

(19)日本国特許庁 (JP)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平10-187527

(43)公開日 平成10年(1998)7月21日

(51)Int.Cl.⁶
G 0 6 F 12/00
9/46

識別記号
5 7 2
3 4 0

F I
G 0 6 F 12/00
9/46

5 7 2 A
3 4 0 F

審査請求 未請求 請求項の数3 FD (全22頁)

(21)出願番号 特願平9-187269

(22)出願日 平成9年(1997)6月30日

(31)優先権主張番号 08/673130

(32)優先日 1996年7月1日

(33)優先権主張国 米国(US)

(71)出願人 591064003

サン・マイクロシステムズ・インコーポレーテッド
SUN MICROSYSTEMS, INCORPORATED
アメリカ合衆国 94303 カリフォルニア州・パロ・アルト・サン・アントニオ ロード・901

(72)発明者 エリック・イー・ハガーステン

アメリカ合衆国・94043・カリフォルニア州・パロ・アルト・コーク・オーク・ウェイ・3451

(74)代理人 弁理士 山川 政樹

最終頁に続く

(54)【発明の名称】 アクセス競合を防ぐ装置および方法

(57)【要約】 (修正有)

【課題】 複数の記憶データ・オブジェクトを有し、複数のスレッドを同時に動作させることができるコンピュータ・システム内でアクセス競合を防ぐ。

【解決手段】 複数の動的ロック構造要素を有する動的ロック構造を備え、また、第1の複数の記憶データ・オブジェクトのうちの第2の複数の記憶データ・オブジェクトをマッピング関数に従って複数の動的ロック構造要素のうちの第1の動的ロック構造要素にマップする。第1の動的ロック構造要素は、第3の複数の記憶データ・オブジェクトの識別表示を記憶するように構成される。第3の複数の記憶データ・オブジェクトは、アクセスされている第2の複数の記憶データ・オブジェクトのサブセットを表し、したがって記憶データ・オブジェクトに現在アクセスしているスレッド以外のスレッドは、動的ロック構造内に記憶されているその識別表示を有する記憶データ・オブジェクトにアクセスすることができない。

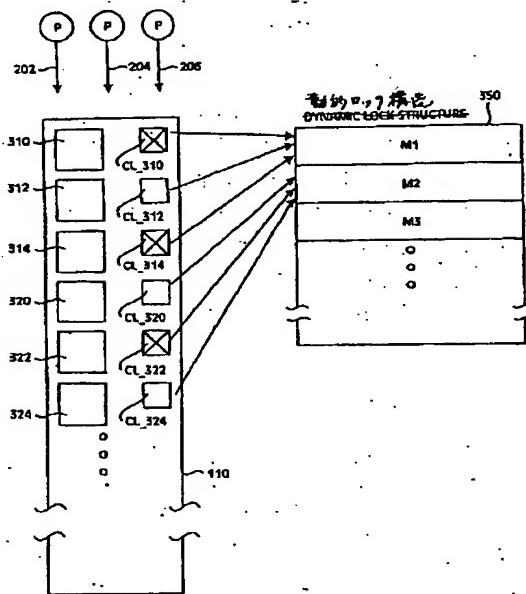


FIG. 3

【特許請求の範囲】

【請求項1】 第1の複数の記憶データ・オブジェクトを有し、複数のスレッドを同時に動作させることができるコンピュータ・システム内で、前記複数のスレッドのうちの2つ以上のスレッドが前記第1の複数の記憶データ・オブジェクトの1つに同時にアクセスすることを許可された場合に起るアクセス競合を防ぐ装置において、

前記第1の複数の記憶データ・オブジェクトの数よりも数が少ない複数の動的ロック構造要素であって、前記第1の複数の記憶データ・オブジェクトのうちの第2の複数の記憶データ・オブジェクトが、マッピング関数に従って前記複数の動的ロック構造要素のうちの第1の動的ロック構造要素にマップし、前記マッピング関数が、前記第1の動的ロック構造要素にマップする前記第2の複数の記憶データ・オブジェクトのうちのただ1つの記憶データ・オブジェクトが前記複数のスレッドのうちの1つのスレッドによって所与の時点においてアクセスされるようにする、複数の動的ロック構造と、

現在アクセスされている前記第2の複数の記憶データ・オブジェクトのサブセットを表す第3の複数の記憶データ・オブジェクトの識別表示を記憶し、それにより現在前記記憶データ・オブジェクトにアクセスしているスレッド以外のスレッドが、前記動的ロック構造内に記憶されているその識別表示を有する記憶データ・オブジェクトにアクセスすることができない、前記第1の動的ロック構造要素に関連する記憶機能とを含んでいるアクセス競合を防ぐ装置。

【請求項2】 第1の複数の記憶データ・オブジェクトを有し、複数のスレッドを同時に動作させることができるコンピュータ・システム内で、前記複数のスレッドのうちの2つ以上のスレッドが前記第1の複数の記憶データ・オブジェクトの1つに同時にアクセスすることを許可された場合に起るアクセス競合を防ぐ方法において、

複数の動的ロック構造要素を有する動的ロック構造を与えるステップと、

前記第1の複数の記憶データ・オブジェクトのうちの第2の複数の記憶データ・オブジェクトをマッピング関数に従って前記複数の動的ロック構造要素の第1の動的ロック構造要素にマップし、それにより前記複数の動的ロック構造要素が前記第1の複数の記憶データ・オブジェクトの数よりも数が少なくなる、マッピングするステップとを含み、前記第1の動的ロック構造要素が、アクセスされている前記第2の複数の記憶データ・オブジェクトのサブセットを表す第3の複数の記憶データ・オブジェクトの識別表示を記憶するように構成され、それにより現在前記記憶データ・オブジェクトにアクセスしているスレッド以外のスレッドが、前記動的ロック構造内に記憶されているその識別表示を有する記憶データ・オブ

ジェクトにアクセスできないようにすることを特徴とする、アクセス競合を防ぐ方法。

【請求項3】 第1の複数の記憶データ・オブジェクトを有し、第1のスレッドと第2のスレッドを同時に動作させることができるとするコンピュータ・システム内で、前記第1のスレッドおよび第2のスレッドが前記第1の複数の記憶データ・オブジェクトのうちの第1の記憶データ・オブジェクトに同時にアクセスすることを許可された場合に起るアクセス競合を防ぐ方法において、

前記第1のスレッドによって試みられた前記第1の記憶データ・オブジェクトへのアクセスを容易にするステップを含み、このアクセスを容易にするステップには、前記動的ロック構造要素が記憶データ・オブジェクトの識別表示を記憶していない場合に、前記第1の記憶データ・オブジェクトの識別表示を前記動的ロック構造の物理ロック部分内に記憶するステップと、

前記動的ロック構造要素が前記第1の記憶データ・オブジェクトと異なる第2の記憶データ・オブジェクトのただ1つの識別表示を前記物理ロック部分内に記憶している場合に、前記第1の記憶データ・オブジェクトの前記識別表示ならびに前記第2の記憶データ・オブジェクトの前記識別表示を前記動的ロック構造の動的リスト部分内に記憶するステップ、および前記動的リストが現在空でないことを示す第1のフラグを前記動的ロック構造の前記物理ロック部分内に記憶するステップと、

前記動的ロック構造要素が前記第1の記憶データ・オブジェクトと同じ第2の記憶データ・オブジェクトのただ1つの識別表示を記憶している場合に、前記第1のスレッドによって試みられた前記第1の記憶データ・オブジェクトへの前記アクセスを失敗したものとして拒否するステップと、

前記物理ロックが、前記動的リストが現在空でないことを示す前記第1のフラグを記憶している場合に、前記第1の記憶データ・オブジェクトの前記識別表示がすでに前記動的リスト内に記憶されているかどうかを確認するために前記動的リストを調べるステップとを含んでおり、この動的リストを調べる前記ステップが、前記動的リストが既に前記第1の記憶データ・オブジェクトの前記識別表示記憶している場合に、前記第1のスレッドによって試みられた前記第1の記憶データ・オブジェクトへの前記アクセスを失敗したものとして拒否するステップと、

前記動的リストがまだ前記第1の記憶データ・オブジェクトの前記識別表示記憶していない場合に、前記第1のスレッドによって試みられた前記第1の記憶データ・オブジェクトへの前記アクセスを許可し、前記第1の記憶データ・オブジェクトの前記識別表示を前記動的リスト内に記憶するステップとを含んでいることを特徴とする方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、コンピュータ・システム内の記憶データ・オブジェクトを共用する方法および装置に関し、さらに詳細には、本発明は、複数のプロセスまたはスレッドが複数の記憶されたデータ・オブジェクトに効率的かつ正確にアクセスすることができる方法および装置に関する。

【0002】

【従来の技術】現代のコンピュータ・システムでは、複数のプロセスまたはスレッドが書き込みまたは読み取りを行うために同じ記憶データ・オブジェクトにアクセスしたいことがある。ここで使用するソフトウェア・プロセスまたはスレッドなる語は、同じ意味であり、一般に論理上別個の実行ストリームをさす。例として、同時に動作している複数の機能は、異なるプロセスまたはスレッド内で実行されていると言える。

【0003】一般に、単一のプロセッサ、複数のプロセッサ、またはコンピュータ・ネットワーク内の複数のコンピュータを介して複数のスレッドを実行することができる。説明のために、図1に、複数のスレッドがプロセッサ102を介して多重方式で実行される多重プログラミング状況の一例を示す。スレッドから見ると、スレッドは、1つの多重プロセッサ内ではなく異なるプロセッサ内で実行されるように見える。

【0004】図1には、図1のコンピュータ・システムに関連し、データを記憶する記憶スペースを示す記憶機能110が示されている。記憶機能(SF, storage facility)110は、例えば、半導体メモリ、フラッシュメモリ、仮想メモリ、磁気記憶装置または光記憶装置など、データを記憶することができる任意の装置または複数の装置の組合せを表す。

【0005】記憶機能110内には、複数の記憶されたデータ・オブジェクト112、114および116が示されている。これらの記憶データ・オブジェクト112、114および116のそれぞれは、スレッド104、106または108の1つが書き込みまたは読み取りを行いたいデータ記憶装置のユニットを表す。各記憶データ・オブジェクトは、スレッドによって実施される特定のタスクに適合するように幾通りもの形で構成され、配置される。例として、スレッド104、106および108がデータベース内のデータを操作するプロセスを表す場合、記憶データ・オブジェクト112は、例えば、スレッド104～108のいずれかが書き込みまたは読み取りを行うことができるデータベースを表す。もちろん、任意の数の記憶データ・オブジェクトを記憶機能110内に準備することができる。データベース内には、例えば、スレッドがアクセスすることができるレコードを表す何百万個以上の記憶データ・オブジェクトがある。

【0006】また、スレッドによって操作すべきデータを記憶するユニットを表すのに「オブジェクト」なる語

を使用しているが、現代の文献においてソフトウェアにおいて「オブジェクト指向」手法と通常呼ばれるものに本発明を限定する意図はないことを念頭に置かれたい。実際、本発明は、これに限定されないが、上述のオブジェクト指向手法を含む従来のプログラミング手法を使用して実施されるものである。

【0007】上述のように、複数のスレッドは、複数のプロセッサを介して実施できる。図2および図3に、複数のスレッドが複数のプロセッサ(図2)上で実行される2つの異なる多重処理状況、および複数のコンピュータ・システム(図3)上で実行される2つの異なる多重処理状況を示す。図2には、それぞれ記憶データ・オブジェクト112、114および116にアクセスするスレッド130、132および134の別の1つを実行している複数のプロセッサ122、124および126を有するコンピュータ・システム120が示されている。図3には、スレッド150、152および154が異なるコンピュータ・システム160、162および164上で実行される別形態が示されている。各スレッド150、152および154は、記憶機能110の任意の記憶データ・オブジェクト112、114および116にアクセスする。

【0008】2つ以上のスレッドが同じ記憶データ・オブジェクトに対してアクセスを試みることがなければ、アクセス競合は起こらない。複数のスレッドが同じ記憶データ・オブジェクト、例えば図2の記憶データ・オブジェクト112に同時にアクセスしたい場合、記憶データ・オブジェクト112の同時アクセスは、誤った結果をもたらす。説明のために、スレッド130が記憶データ・オブジェクト112内のデータを更新したい状況について考える。スレッド130が記憶データ・オブジェクト112の内容を更新している間、他のスレッド132がそれを読み取るために記憶データ・オブジェクト112にアクセスすることを許可された場合、アクセス競合が起こる。これは、スレッド132が記憶データ・オブジェクト112の一部更新されたコピーしか得ることができないためである。したがって、アクセス競合のために、スレッド132は誤ったデータを受け取ることになる。

【0009】従来技術において、上述のアクセス競合問題を回避するいくつかの技法が提案されている。図4に、複数のスレッドが单一の記憶データ・オブジェクトに同時にアクセスすることを許可された場合に起こるアクセス競合を回避する従来技術の技法を示す。次に図4を参照すると、記憶機能110内の記憶データ・オブジェクト180、182、184、186、188および189にアクセスすることができるスレッドを表す複数のスレッド202、204および206が示されている。アクセス競合を回避するために、各記憶データ・オブジェクト180～189にはロッキング・フラグが関

連付けをされている。例えば、それぞれの記憶データ・オブジェクト180、182、184、186、188および189に対応するロッキング・フラグ190、192、194、196、198および199が示されている。

【0010】スレッド、例えばスレッド202は、記憶データ・オブジェクト、例えば記憶データ・オブジェクト180にアクセスした場合、アクセスした記憶データ・オブジェクトに対応するロッキング・フラグをセットする。記憶データ・オブジェクト180の場合、スレッド202より記憶データ・オブジェクト180がアクセスされたとき、対応するロッキング・フラグ190がセットされる。ロッキング・フラグ190がセットされている間、他のスレッド、例えばスレッド204または206は、記憶データ・オブジェクト180にアクセスすることができない。スレッド202は、そのアクセスを終了すると、ロッキング・フラグ190をリセットし、それにより他のスレッドが記憶データ・オブジェクト180にアクセスすることができるようになる。

【0011】アクセス競合を回避する従来技術の技法は、比較的少数の記憶データ・オブジェクトに対しては十分に役立つ。しかしながら、多数の記憶データ・オブジェクト、例えば何百万もの記憶データ・オブジェクトを有するシステムにおいては、すべての記憶データ・オブジェクトに対するフラグの要求は、多数の記憶データ・オブジェクトに対して何百万個以上ものフラグが必要となるので、記憶機能資源の極めて非能率的な使用を呈する。各フラグが、例えば、フラグの読み取りまたは書き込みの速度を高めるために通常行われるように、1バイト（すなわち8ビット）または1ワード（すなわち32ビット）を必要とする場合、すべての従来技術のフラグを実施するのに必要なビットの数は法外に多くなる。各フラグが1ビットしか必要としない場合でも、そのような多数のフラグは、システムによっては、アクセス競合を防ぐためだけに容認できないほど大きい記憶機能オーバヘッドとなる。

【0012】さらに、記憶データ・オブジェクト当たりのビットの数は、システムによってはあまり多くないこともある。この場合、実際のデータ記憶に使用されるビットの数に対するフラグを実施するのに必要なビットの数の比率は、容認できないほど大きくなる。このオーバヘッド率はまた、理解できるように、記憶データ・オブジェクトのサイズが縮小するにつれて増大する傾向がある。

【0013】

【発明が解決しようとする課題】以上のことから、複数のスレッドが同じ記憶データ・オブジェクトにアクセスしようと試みた場合に起こるアクセス競合を回避する改善された方法および装置が必要である。

【0014】

【課題を解決するための手段】本発明は、一実施形態では、第1の複数の記憶データ・オブジェクトを有し、複数のスレッドを同時に動作させることができるコンピュータ・システム内でアクセス競合を防ぐ装置に関する。アクセス競合は、複数のスレッドのうちの2つ以上のスレッドが第1の複数の記憶データ・オブジェクトの1つに同時にアクセスすることを許可された場合に起こる。

【0015】本装置は、複数の動的ロック構造要素を有する動的ロック構造を含んでいる。複数の動的ロック構造要素は、第1の複数の記憶データ・オブジェクトの数よりも数が少ない。第1の複数の記憶データ・オブジェクトのうちの第2の複数の記憶データ・オブジェクトは、マッピング関数に従って複数の動的ロック構造要素のうちの第1の動的ロック構造要素にマップする。マッピング関数では、第1の動的ロック構造要素にマップする第2の複数の記憶データ・オブジェクトのうちのただ1つの記憶データ・オブジェクトが、複数のスレッドのうちの1つのスレッドによって所与の時点においてアクセスされるようになっている。

【0016】また、第3の複数の記憶データ・オブジェクトの識別表示（アイデンティティ）を記憶する第1の動的ロック構造要素に関連付けをされた記憶機能が含まれている。第3の複数の記憶データ・オブジェクトは、現在アクセスされている第2の複数の記憶データ・オブジェクトのサブセットを表し、したがって記憶データ・オブジェクトに現在アクセスしているスレッド以外のスレッドは、動的ロック構造内に記憶されているその識別表示を有する記憶データ・オブジェクトにアクセスすることができない。

【0017】他の実施形態では、第1の複数の記憶データ・オブジェクトを有し、複数のスレッドを同時に動作させることができるとするコンピュータ・システム内でアクセス競合を防ぐ方法に関する。この方法は、複数の動的ロック構造要素を有する動的ロック構造を与えるステップを含む。また、第1の複数の記憶データ・オブジェクトのうちの第2の複数の記憶データ・オブジェクトをマッピング関数に従って複数の動的ロック構造要素のうちの第1の動的ロック構造要素にマップするステップがある。マッピング関数のために、複数の動的ロック構造要素は、第1の複数の記憶データ・オブジェクトの数よりも数が少なくなる。第1の動的ロック構造要素は、第3の複数の記憶データ・オブジェクトの識別表示を記憶するように構成される。第3の複数の記憶データ・オブジェクトは、アクセスされている第2の複数の記憶データ・オブジェクトのサブセットを表し、したがって記憶データ・オブジェクトに現在アクセスしているスレッド以外のスレッドは、動的ロック構造内に記憶されているその識別表示を有する記憶データ・オブジェクトにアクセスすることができない。

【0018】本発明の上記その他の利点は、以下の詳細

な説明を読み、各図面を検討すれば明らかになろう。

【0019】

【発明の実施の形態】特に、複数のプロセスまたはスレッドが複数のデータ・オブジェクトに効率的かつ正確にアクセスすることができる本発明について説明する。以下の説明は、本発明を完全に理解することができるように、多数の特定の詳細を示す。しかしながら、本発明は、これらの特定の詳細の一部またはすべてを用いなくとも実施できることが当業者には明らかであろう。他の場合には、本発明を不必要に曖昧にしないために、周知の構造およびプロセス・ステップについては詳細に説明していない。

【0020】本発明の一態様によれば、従来必要とされた全ての記憶データ・オブジェクトに対してのロック・フラグの要求をなくすせるので有利である。所与の時点においてアクセスされている記憶データ・オブジェクトの実際の数は、所与の記憶機能内の記憶データ・オブジェクトの総数よりも少ないと理解されたい。この理解があれば、所与の時点において、どのスレッドからもアクセスされない記憶データ・オブジェクトが多数ある（したがって、アクセス競合の危険がない）ので、すべての記憶データ・オブジェクトに対して記憶機能内にロックを物理的に準備する必要はないということである。しかしながら、課題は、アクセス競合を防ぐために物理的に実施されるロックの数を減らす（それにより、この目的に充てられた記憶機能オーバヘッドを減らす）と同時に、スレッドから記憶データ・オブジェクトにアクセスすることができ、他のスレッドを迅速かつ効率的にロック・アウトすることである。

【0021】本発明の一態様によれば、スレッドにより現在アクセスされている記憶データ・オブジェクトをロック・アウトする複数の物理ロックが提供される。本願で使用している物理ロックなる語は、一般にメモリ内で実際に実行されるデバイスをさし、その実行には、いくらかの記憶機能資源が必要である。本発明に従って提供される物理ロックの数は、図4の従来技術の手法と異なり、使用できる記憶データ・オブジェクトの数よりも数が少ないことが有利である。例えば、あるシステムでは、1000個の物理ロックを使用して、何百万個の記憶データ・オブジェクトへのアクセスを追跡する何百万個の概念ロックを実施することもある。

【0022】概念的に言えば、すべての記憶データ・オブジェクトには論理ロックすなわち概念ロックが関連付けられている。所与のシステムに対しての本発明の概念ロックの数は、従来技術の実際の物理ロックと同数であるが、本発明による概念ロックは、実行にあたり実際の記憶機能資源を必要としないことが有利である。この明細書でのそれらの説明は、本発明を当業者に周知の用語で示すものである。概念ロックは、実世界のコンピュータ・システム内で実行する必要はないことを常に念頭に

置かれたい。

【0023】本発明の特徴および利点の詳細な説明のために図5を参照する。図5には、複数の記憶データ・オブジェクト310、312、314、320、322および324を記憶する記憶機能110が示されている。特定の記憶データ・オブジェクトがスレッドによって現在アクセスされているかどうか（したがって、他のスレッドによって試みられたアクセスをロック・アウトすべきかどうか）を概念的に示すために、それぞれの記憶データ・オブジェクト310、312、314、320、322および324に関連付けをされた複数の概念ロックCL_310、CL_312、CL_314、CL_320、CL_322およびCL_324が示されている。例えば、概念ロックCL_310は、ロック状態にあり、スレッド、例えばスレッド202が記憶データ・オブジェクト310に現在アクセスしており、他のスレッド、例えばスレッド204または206によって試みられる記憶データ・オブジェクト310へのアクセスを拒否すべきことを示すものとして示されている。同様に、概念ロックCL_314は、ロック状態にあり、それにより現在その対応する記憶データ・オブジェクト314にアクセスしているスレッドを除いて、他のスレッドによってその関連付けをされた記憶データ・オブジェクト314へのアクセスを防ぐものとして示されている。一方、概念ロックCL_312、CL_320、CL_322およびCL_324は、非ロック状態にあり、スレッド202、204および206のいずれかがそれらに関連付けをされた記憶データ・オブジェクトへのアクセスを要求することができ、それらの要求は直ちに許可できることを示すものとして示されている。

【0024】所与の時点においてロックされた概念ロックを追跡するために、本発明の一態様によれば、動的ロック構造350が提供される。動的ロック構造350の役目は、アクセス競合を防ぐために記憶データ・オブジェクトへのアクセスを管理することである。概念ロックと異なり、動的ロック構造350およびその構成部分は、実際には組み込まれたメモリである。すなわち、動的ロック構造350を実行する場合、実際には実際の物理記憶機能資源が使用される。動的ロック構造350は、スレッドによって現在アクセスされている概念ロックの識別表示（これは一実施形態において記憶データ・オブジェクトの識別表示を表す）のみを追跡することが必要であるので、その実施に充てられた記憶機能資源の総量は、従来技術において必要とされる量、例えば、従来技術図4のロック・フラグを実施するのに必要なメモリの量と比較して少ないことが有利である。

【0025】図5に示すように、動的ロック構造350は複数の構造要素を含んでおり、そのうちの3つの要素M1、M2およびM3が示されている。本発明の一態様によれば、動的ロック構造350内の動的ロック構造要

素の数は、概念ロックの数よりも少ないことが有利である。動的ロック構造要素の数を少なくするために、複数の概念ロックが、何らかのマッピング関数に従って各動的ロック構造要素にマップされる。適切なマッピング関数により、同時に使用されている概念ロックが異なる動的ロック構造要素にマップされるようになり、同時に迅速に実行されることが好ましい。システムは、例えば、記憶データ・オブジェクトのアドレスの中央部から（例えば、適切なビット・マスクおよびシフト命令を使用して）ビット群を選択するか、あるいは従来のハッシュ関数を使用することができる。図5の例では、CL_310からCL_319までのすべての概念ロックは動的ロック構造要素M1にマップし、CL_320からCL_329までのすべての概念ロックは動的ロック構造要素M2にマップする、などとなる。

【0026】図6および図7に、動的ロック構造要素M1、M2およびM3を含む動的ロック構造350の一部を本発明の一実施形態に従って詳細に示す。各動的ロック構造要素、例えば動的ロック構造要素M1内には、物理ロック部分、リストロック・フラグ部分および動的リスト部分が示されている。一実施形態では、物理ロック部分は、対応する動的構造要素にマップする1つの概念ロックの識別表示か、またはこの動的ロック構造要素に関連付けをされた動的リストが空かどうかを示すフラグを記憶するメモリ構造を含んでいる。概念ロックは概念としてのみ存在するので、概念ロックの識別表示は、一実施形態では、その対応する記憶データ・オブジェクトの識別表示によって表されることを理解されたい。記憶データ・オブジェクトは、例えば、メモリ内に記憶されている場合、その開始アドレスによって識別される。物理ロック部分内に記憶されるフラグEMPTY及びNOT_EMPTYは、一実施形態では、单一ビット、例えば、单一ビットの0と1の値によって実行される。或いは、他の従来のロック実施形態を使用することもできる。

【0027】図7の動的ロック構造要素M1を参照すると、例えば、物理ロック410は、動的ロック構造要素M1に関連付けをされた動的リスト412が空でないことを示すフラグNOT_EMPTYを記憶するものとして示されている。一方、動的ロック構造要素M3の物理ロック430は、動的ロック構造要素M3の対応する動的リスト432が現在空であることを示すフラグEMPTYを記憶するものとして示されている。

【0028】動的ロック構造要素にマップされる概念ロックは、その動的ロック構造要素にマップするすべての概念ロックの中でスレッドによって現在アクセスされている唯一の概念ロックであるならば、その概念ロック識別表示が、その動的ロック構造要素の物理ロック部分内に記憶される。例として、動的ロック構造要素M2にマップするすべての概念ロックの中で概念ロックCL_3

22のみが現在アクセスされている場合、概念ロックCL_322の識別表示は、動的ロック構造要素M2の物理ロック部分、すなわち物理ロック部分420内に記憶されることが好ましい。後で詳細に説明するように、概念ロック識別表示が動的ロック構造要素の物理ロック部分内に記憶できる場合は常に、スレッドが記憶データ・オブジェクトにアクセスでき、本発明ではこのことを利用して速度を高め、概念ロックを獲得したり、解放することができる。

- 10 【0029】動的ロック構造350の各動的ロック構造要素はさらにリストロック・フラグを含んでいる。リストロック・フラグは、それに関連付をされた動的リスト、すなわち同じ動的ロック構造要素に属する動的リストが所与の時点においてアクセス可能であるかどうかを示すために使用される。図7の実施形態では、リストロック・フラグは、LOCK_LISTおよびUNLOCK_LISTの2つの状態を有し、一実施形態では、单一ビット、例えば、ある单一ビットの0と1の値によって実行される。例として、動的ロック構造要素M1は、「LOCK_LIST」状態を現在記憶しており、対応する動的リスト412へのアクセスが現在許可されていないことを示すリストロック・フラグ414を有するものとして示されている。

- 20 【0030】本発明の一態様によれば、各動的ロック構造要素の動的リスト部分は、概念ロック識別表示の動的に生成されたリストを含んでいる。動的リスト内に記憶されている概念ロック識別表示は、その動的リストに対応する動的ロック構造要素にマップし、それらに関連付けをされた記憶データ・オブジェクトがアクセスされているので現在「ロック」されている概念ロックの識別表示を表す。本発明の一態様によれば、動的に生成されたリスト、例えば、リンクされたリスト・データ構造を使用して概念ロック識別表示を記憶することにより、記憶機能の使用がフレキシブルかつ有利に最小限に抑えられる。

- 30 【0031】動的リストは、例えば、それぞれ動的ロック構造要素M2およびM3の動的リスト422および432の場合に示されるように、空であり得る。動的ロック構造要素内の空の動的リストに関連付けられた物理ロック部分は、（動的ロック構造要素に現在1つの概念ロック識別表示しか記憶されていない場合）その関連付けられた動的ロック構造要素にマップされて現在ロックされている1つの概念ロックの識別表示を記憶するか、または、（動的リスト部分にも動的ロック構造要素の物理ロック部分にも概念ロック識別表示が記憶されていない場合）フラグEMPTYを記憶する。概念ロック識別表示を記憶している物理ロックおよびEMPTYフラグを記憶している物理ロックは、動的ロック構造要素M2の物理ロック420および動的ロック構造要素M3の物理ロック430に示されている。

【0032】或いは、動的リストは、複数の概念ロック識別表示を含んでいることもある。上述のように、動的ロック構造要素が1つの概念ロック識別表示しか記憶していない場合、その概念ロック識別表示は、アクセス速度を高めるために、動的ロック構造要素の物理ロック部分内に記憶されることが好ましい。一方、動的ロック構造要素が複数の概念ロック識別表示、すなわちこの動的ロック構造要素に何れもマップされてあるスレッドによってアクセスされている複数の概念ロックの識別表示を記憶している場合、これら複数の概念ロック識別表示はすべて動的リスト部分内に記憶されることが好ましい。したがって、動的リスト部分は、2つ以上の概念ロック識別表示を含んでいるか、または概念ロックの識別表示を1つも含んでいないかのどちらかである。動的ロック構造要素の動的リスト部分が空でない場合、その動的ロック構造要素に関連付けられた物理ロック部分は、NOT_EMPT_Yフラグを記憶していることが好ましい。図7を参照すると、NOT_EMPT_Yフラグは、あるスレッドによってアクセスされ、動的ロック構造要素M1にマップしている2つの記憶データ・オブジェクトに対応する2つの概念ロック識別表示を記憶するために動的リスト412が使用される場合は、動的ロック構造要素M1の物理ロック410部分内に記憶されるものとして示されている。

【0033】動的ロック構造要素の物理部分内にフラグNOT_EMPT_Yが存在する場合は、スレッドには、概念ロックを獲得したり解放をしようとする場合に、動的リストを調べるべき指示が与えられると有利である。一方、動的ロック構造要素の物理部分内にフラグEMPT_Yが存在する場合、スレッドには、この動的ロック構造要素にマップする概念ロックのいずれも現在アクセスされていないこと、すなわちそれらのいずれでも獲得できることが指示されると有利である。動的ロック構造要素の物理部分内にある概念ロック識別表示が存在する場合、この動的ロック構造要素にマップするすべての概念ロックのうち、この概念ロック識別表示のみが現在保持されていること、そして、もし概念ロック識別表示が重要でなければ、動的リストが空となるだろう（したがって、この動的ロック構造にマップする他のすべての概念ロックが必要ならば獲得できる）から動的リストを調べる必要はないことを、スレッドに指示すると有利である。したがって（同時に保持されている概念ロックが異なる動的ロック構造要素にマップする限り、すなわち適切に選択したハッシュ関数によってもたらされ得る状況における限り）物理ロック部分のみを使用して、概念ロックを容易に獲得したり、解放することができる。

【0034】本発明の動的ロック構造の動作および本発明のアクセス競合回避技術におけるその役割は、図8、図9、図10および図11を参照すればよりよく理解できよう。図8および図10は、本発明の一態様によれ

ば、スレッド、例えば図5のスレッド202が記憶データ・オブジェクトにアクセスしたい場合にアクセス競合が起らぬないようにするためにとられるステップを示す。図9および図11は、それぞれ図8および図10のステップの説明をさらに助ける例を示す。

【0035】図8は、本発明の一態様によれば、所与のスレッドが記憶データ・オブジェクトに関連付けられた概念ロックを獲得したい場合に、その概念ロックが保持されている間、他のスレッドがその記憶データ・オブジェクトにアクセスするのを防ぐために行うステップを示す。ステップ510において、概念ロックXが保持されている間、他のスレッドが概念ロックXに関連付けられた記憶データ・オブジェクトにアクセスするのを防ぐために、所与のスレッド、例えばスレッドAは、所与の概念ロック、例えば概念ロックXを獲得することを望む。

【0036】概念ロックXがマップされる動的ロック構造要素の動的リストが空であり、かつ関連付けられた物理ロックが他の概念ロック識別表示を記憶するのに使用されていない場合、概念ロックXの識別表示をこの動的ロック構造要素の物理ロック部分内に記憶する（ステップ512）。概念ロックXの識別表示の物理ロック部分内への記憶が成功した場合には、ステップ514に進み、そこで概念ロックXが獲得され、スレッドAが概念ロックXを保持している間、他のスレッドが概念ロックXに関連付けられた記憶データ・オブジェクトにアクセスできないことを示すメッセージを、概念ロックXを獲得しないスレッド、例えばスレッドAに戻す。

【0037】一方、関連付けられた動的リストが空でない（したがって、関連付けられた物理ロックがNOT_EMPT_Yフラグを記憶している）場合、または関連付けられた物理ロックが他の概念ロックの識別表示を記憶するのに使用されている場合には、ステップ516に進み、そこで関連付けられた動的リストをロック・アウトし、スレッドAが関連付けられた動的リストの操作を開始できるようにする。概念ロックXがマップされる動的ロック構造要素が1つまたは複数の概念ロック識別表示を現在保持している場合にのみ、ステップ516に進むことに留意されたい。他のスレッドが関連付けられた動的リストを現在ロック・アウトしている場合は、ステップ516において、関連付けられた動的リストを得られ、それをロック・アウトするまで待ち、その後図8の後続のステップに進むことが好ましい。

【0038】ステップ517において、物理ロック内に保持されている概念ロック識別表示があればそれを一次記憶変数TEMP内に保存する。さらに、物理ロックの内容を、この動的ロック構造に関連付けられた動的リストが図8のステップの終了時に空でないことを示すフラグNOT_EMPT_Yにセットする。ステップ517では、スレッドAが概念ロックXを獲得できる場合、物理ロック部分内に保持された概念ロック識別表示の動的リ

スト内への後続の記憶を容易にすると有利である。図5において後で理解できるように、概念ロックXがすでに他のスレッドによって保持されており、スレッドAがアクセスすることができない場合は、後続のステップにおいてステップ517を逆転させ、物理ロック内に記憶された値をも元に戻し、その後ロックXを獲得する試みが失敗したことをスレッドAに知らせる。

【0039】動的リストの検査に進む前に、ステップ540において、フラグEMPTYを含んでいるかどうかを確認するために位置TEMPを検査する。例えば、スレッドAがステップ512を終了した後でスレッドAがステップ516を実行する前に、他のスレッドが動的ロック構造要素の物理ロック部分内に記憶されている最後の概念ロックをロック解除している場合、プロセス中のこの点においては、TEMP位置には、フラグEMPTYが含まれている。

【0040】位置TEMPがフラグEMPTYを含んでいる（とステップ540において確認された）場合は、ステップ542に進み、概念ロックXの識別表示を物理ロック部分内に記憶する。その後で、ステップ544に進み、そこでステップ516のアンドゥ（解除）を行う、すなわち他のスレッドがこの動的リストにアクセスできるように関連付けられた動的リストをロック解除する。ステップ514において、概念ロックXを得ることができ、スレッドAが概念ロックXを保持している間、他のスレッドが概念ロックXに関連付けられた記憶データ・オブジェクトにアクセスできないことを示すメッセージを、概念ロックXを獲得したいスレッド、例えばスレッドAに戻る。理解できるように、ステップ540、542、544および514を通るバスは、スレッドAが概念ロックXを獲得できるようにするための関連付けられた動的リストの操作や検査が不要なので最適化されている。

【0041】位置TEMPがフラグEMPTYを含んでいない（とステップ540において確認された）場合、ステップ518に進み、概念ロックXが使用できるかどうかを確認するために、動的リストならびにこの動的ロック構造要素に関連付けられた位置TEMPを検査する。概念ロックXは、他のスレッドによって現在アクセスされており、したがってその識別表示がすでに関連付けられた動的リスト内または（スワップ・ステップ517において物理ロックからその値を得る）TEMP位置に記憶されている場合には、使用できない。使用できない場合、ステップ519に進み、そこで前に物理ロック部分内に記憶された値をも物理ロック部分に戻すことによって前のステップ517をアンドゥ（解除）する。次いで、ステップ520に進み、他のスレッドが関連付けられた動的リストにアクセスできるよう、関連付けられた動的リストをロック解除する。その後、ステップ522において、概念ロックXを獲得する試みが失敗したこ

とを知らせるメッセージをスレッドAに戻す。

【0042】一方、ステップ518において、概念ロックXが使用できる（概念ロックXの識別表示が関連付けられた動的リスト内にもTEMP位置にも記憶されていない）ことが確認された場合、ステップ524に進み、そこで概念ロックXの識別表示を含めて、この動的ロック構造にマップし、スレッドによって現在保持されているすべての概念ロックIDを、関連付けられた動的リスト内に記憶する。これで、ステップ524を実行した後、概念ロックXに関連付けられた動的ロック構造にマップし、現在保持されている複数の概念ロック識別表示が関連付けられた動的リスト内に記憶されていることになる。

【0043】次いで、ステップ528において、他のスレッドが関連付けられた動的リストにアクセスできるよう、この関連付けられた動的リストをロック解除する。次いで、ステップ530において、概念ロックXを獲得する試みが成功したことを知らせるメッセージをスレッドAに戻し、これで概念ロックが保持され、したがって他のスレッドはそれにアクセスすることができない。

【0044】図9は、一例において、スレッド202が図7の概念ロックCL_322を獲得したいときの図8のステップを示す（ステップ550）。概念ロックCL_322は動的ロック構造要素M2にマップするので、ステップ522において、物理ロック420の値を（実施形態では値0によって表される）フラグEMPTYと比較し、物理ロック420の内容がEMPTYに等しければ、概念ロックCL_322の識別表示を物理ロック420の内容とスワップする。

【0045】概念ロックCL_322がマップする動的ロック構造要素、すなわち動的ロック構造要素M2が、その関連付けられた概念ロック、例えば図5の概念ロックCL_320、CL_322およびCL_324の識別表示を記憶していない場合は、物理ロック420の内容は、EMPTYに等しくなる。物理ロック420がEMPTYフラグを記憶していることがステップ522において確認された場合は、比較およびスワップ操作は成功であり、ステップ553に進み、そこで概念ロックCL_322の獲得の試みが成功したことをスレッド202に知らせる。ステップ533の後、概念ロックCL_322の識別表示を物理ロック420内に記憶する。

【0046】一方、ステップ522において、物理ロック420の内容がフラグEMPTYに等しくない（物理ロック420が動的ロック構造要素M2に関連付けられた動的リスト、すなわち動的リスト422が空でないことを示すフラグ状態NOT_EMPTYを記憶しているか、または物理ロック420が概念ロックの識別表示を現在記憶している）ことが確認された場合、ステップ554に進み、そこでリストロック・フラグ424を状態LOCK_LISTにセットする。ステップ554にお

いて、動的リスト422にアクセスしようと試みる他のスレッドをロック・アウトすることによって、スレッド202のために動的リスト422を獲得する。一実施形態では、LOCK_LISTは、単一ビットの値1だけで表される。

【0047】リストロック・フラグ424をLOCK_LISTにセットすることにより、他のスレッドからの妨害なしにスレッド202のために動的ロック構造M2に関連付けられた動的リストを操作することができる。ステップ556において、一次記憶位置TEMPをフラグNOT_EMPTYにセットし、この一時記憶位置TEMPを物理ロック420にスワップする。

【0048】ステップ558において、概念ロックCL_322の識別表示がすでに動的リスト422内かまたは一次位置TEMP内に記憶されているかを確認する。概念ロックCL_322の識別表示がすでに動的リスト422内かまたは一次位置TEMP内に記憶されている場合、他のスレッドがすでに概念ロックCL_322を獲得している。この場、概念ロックCL_322を獲得するスレッド202による試みは失敗であり、ステップ560に進み、一次位置TEMPの識別表示を物理ロック420内に戻す、即ち前のスワップ・ステップ556を解除する。

【0049】ステップ562において、リストロック・フラグ424をUNLOCK_LISTにセットする、すなわち他のスレッドが動的リスト422にアクセスすることができるよう、前のロッキング・ステップ554を解除する。ステップ564において、概念ロックCL_322を獲得する試みが失敗したことを知らせるメッセージをスレッド202に戻す。

【0050】一方、ステップ558において、所望の概念ロック、すなわち概念ロックCL_322の識別表示が動的リスト422内かまたは一次記憶位置TEMP内に記憶されていないことが確認された場合、概念ロックCL_322は、他のスレッドによって獲得されておらず、したがってスレッド202が獲得することができる。この場合は、ステップ566に進み、動的リスト422に概念ロックCL_322の識別表示を追加する。

【0051】ステップ570において、以前のロッキング・ステップ554を解除するために、リストロック・フラグ424を状態UNLOCK_LISTにセットし、それにより他のスレッドが動的リスト422にアクセスすることができるようとする。ステップ572において、概念ロックCL_322を獲得するスレッド202の試みが成功したことを知らせるメッセージをスレッド202に戻す。

【0052】ステップ580において、位置TEMPがEMPTYに等しいことかどうかを検査する。位置TEMPがEMPTYに等しければ、ステップ582、584および553を含む高速バスを介して概念ロックCL

_322の獲得を開始する。ステップ582～584は、前に図8に関して説明したステップ542～544に類似している。その後、ステップ553に進み、概念ロックCL_322を獲得する試みが成功したことをスレッド202に知らせる。

【0053】図9において、スレッド202は、異なる3つのバスを介して概念ロックCL_322を獲得できることに留意されたい。動的ロック構造要素M2が前に概念ロック識別表示を記憶しておらず、かつ概念ロックCL_322がこの動的ロック構造要素M2によって記憶された第1の概念ロックを表す場合、ステップ550、552および553に進み、スレッド202が概念ロック322に迅速にアクセスすることができるようになる。これらのステップは、数が少ないだけでなく、ステップ552におけるスワップ操作が一般にプロセッサによって実行される高速操作の1つであるので比較的迅速である。この場合、本発明のアクセス競合回避システムの性能は、すべての單一記憶データ・オブジェクトのロック・フラグを記憶するために相当量の記憶機能オーバヘッドを必要とする従来技術の方法とほとんど同じくらい効率的である。

【0054】一方、ステップ552とステップ554の間で競合状態が起っている場合、関連付けられた動的リストの操作または検索を必要とせずに、比較的速いバス(550/552/554/556/580/382/584/443)を介して概念ロックの獲得を達成することができる。

【0055】一方、動的ロック構造要素M2が既に1つまたは複数の概念ロック識別表示、例えば概念ロックCL_320、CL_322およびCL_324の1つを保持している場合、ステップ550、552、554、556、558、566、568、570および572を含むバスに進み、スレッド202が概念ロックXを獲得することができるようになり、またこれら複数の概念ロック識別表示を動的ロック構造要素M2の動的リスト内に記憶することができるようになる。

【0056】図10に、本発明の一態様に従って、スレッド、例えばスレッドAが、前に獲得した概念ロック、例えば概念ロックXを解放したい場合に行うステップ40(ステップ602)を示す。ステップ604において、関連付けられた物理ロックの内容が概念ロックXの識別表示を含んでいる場合、EMPTYフラグを関連付けられた物理ロックの内容内に記憶する。関連付けられた物理ロックの内容が概念ロックXの識別表示である場合、他の概念ロックIDは、概念ロックXがそれにマップする動的ロック構造要素によって記憶される。この場合は、ステップ604において、解放を迅速に行い、ステップ606において、概念ロックXは解放され、他のスレッドが獲得のために使用できると考えられる。

【0057】一方、関連付けられた物理ロックの内容が

概念ロックXの識別表示ではない場合、概念ロックXがマップする動的ロック構造要素は、複数の概念ロック識別表示を現在記憶している。この場合、ステップ608に進み、そこで、他のスレッドが現れて関連付けられた動的リストの修正をするのを防ぐために、この関連付けられた動的リストをロック・アウトすると同時に、スレッドAのために、概念ロックXを、関連付けられた動的リストからの除去によって解放する。

【0058】ステップ610において再び、前のステップ604の検査およびスワップ操作を実施する。ステップ610の検査およびスワップ操作は、ステップ608において動的リストをロック・アウトしようとしている間、スレッドAと現在同時に動作している他のスレッドは、概念ロックXの識別表示を関連する物理ロック内に入ってしまっているので、(ステップ608において)関連付けられた動的リストをロック・アウトした後で行うことが望ましい。他のスレッドが概念ロックXの識別表示を関連付けられた物理ロック内に入る理由は、最適化と関係があり、図10の残りの説明から明らかになろう。

【0059】ステップ610において、関連付けられた物理ロックの内容が概念ロックXの識別表示であり、検査およびスワップ操作が成功したことが確認された場合、ステップ612に進み、そこで他のスレッドが関連付けられた動的リストにアクセスすることができるよう、関連付けられた動的リストをロック解除する、すなわちロック・アウト・ステップ608の逆を行う。ステップ612の実行をした後、概念ロックXはスレッドAによって解放されたと考えられ、他のスレッドが獲得することができる。

【0060】一方、ステップ610において、関連付けられた物理ロックの内容がまだ概念ロックXの識別表示でないこと、すなわち検査およびスワップ操作が再び失敗したことが確認された場合、ステップ614に進み、そこで概念ロックXの識別表示を関連付けられた動的リストから除去する。概念ロックXの識別表示は関連付けられた動的リスト内ではなく、また関連付けられた物理ロックの内容内になかった(とステップ610において判定された)ので、概念ロックXがマップする動的ロック構造要素は、ステップ614を実行した後、概念ロックXの識別表示を追跡しない。したがって、後続のスレッドが現れ、図10のステップが終了した後、概念ロックXを獲得することができる。

【0061】ステップ616、618、620、622および624において、可能なら、概念ロックXがマップする動的ロック構造要素がただ1つの概念ロック識別表示を記憶している場合、その概念ロック識別表示が好ましくは関連付けられた物理ロックの内容内に移動されるよう動的ロック構造要素を最適化する。従って、残りの1つの概念ロックXを保持するスレッドは、それを解

放したい場合、ステップ602、604および606を含むバスを介して迅速にその解放をすることができる。この最適化を行わなければ、フラグNOT_EMPTYにセットされた物理ロックは、フラグNOT_EMPT Yにセットされない。EMPTY状態に戻れば、概念ロックをバス602/604/606を介して迅速に解放することができるので有利である。

【0062】ステップ616において、概念ロックXの識別表示が(ステップ164において)関連付けられた動的リストから削除された後に、関連付けられた動的リストにただ1つの概念ロック識別表示が残されているかどうかを確認する。関連付けられた動的ロック構造要素が追跡すべき複数の概念ロックIDが残っている場合は、ステップ624に進み、そこでスレッドAが概念ロックXを解放するステップを終了する。

【0063】一方、概念ロックXがマップする動的ロック構造要素が、その関連付けられた動的リスト内にただ1つの概念ロック識別表示を追跡する場合は、ステップ616からステップ618へ進み、そこで関連付けられた物理ロックの内容を残りの1つの概念ロックの識別表示にセットする。ステップ620において、関連付けられた動的ロック構造要素がその物理ロック内ならびにその動的リスト内でこの残りの1つの概念ロックを追跡しないように、この残りの1つの概念ロックの識別表示を関連付けられた動的リストから削除する。ステップ622において、後続のスレッドが関連付けられた動的リストにアクセスすることができるよう、関連付けられた動的リストをロック解除する、すなわちロッキング・ステップ608を逆転させる。ステップ624において、概念ロックXを解放する場合に行うステップを終了する。

【0064】図10のステップは、(図7に鑑みて)図11によって示される例を参照すればより完全に理解することができる。図11において、スレッド202は、前に獲得した概念ロックCL_322を解放することを望む(ステップ652)。ステップ654において、物理ロック420の内容が概念ロックCL_322の識別表示を含んでいる場合、EMPTYフラグを(図7の)関連付けられた物理ロック420の内容内に記憶する。

40 関連付けられた物理ロック420の内容が概念ロックCL_322の識別表示である場合、他の概念ロックIDは、概念ロックCL_322がマップする動的ロック構造要素、すなわち動的ロック構造要素M2によっては記憶されない。この場合、ステップ654において、解放を迅速に実施し、概念ロックCL_322が解放され、ステップ656において、他のスレッドが獲得のために使用できると考えられる。

【0065】一方、関連付けられた物理ロック420の内容が概念ロックCL_322の識別表示を含んでいる場合、動的ロック構造要素M2は現在、複数の概念ロック

ク識別表示（現在保持されている概念ロックCL_32の他に少なくとも2つ以上）を記憶していることが理解される。この場合、ステップ658に進み、そこで他のスレッドが現れて関連付けられた動的リスト422の修正をするのを防ぐために、この関連付けられた動的リスト422をロック・アウトすると同時に、スレッド202のために、概念ロックCL_32を、関連付けられた動的リスト422からの除去によって解放する。

【0066】ステップ660において、は再び、前のステップ654の検査およびスワップ操作を実施する。ステップ660の検査およびスワップ操作は、ステップ658において関連付けられた動的リスト422をロック・アウトしようとしている間に、スレッド202と現在同時に動作している他のスレッドは、概念ロックCL_32の識別表示を関連付けられた物理ロック420内に入れてしまっているので、（ステップ658において）関連付けられた動的リスト422をロック・アウトした後で行うことが望ましい。他のスレッドが概念ロックCL_32の識別表示を関連付けられた物理ロック420内に入れる理由は、図10のステップ616、618、620、622および624に関して前に論じた。

【0067】図11のステップ660において、関連付けられた物理ロック420の内容が概念ロックCL_32の識別表示であり、検査およびスワップ操作が成功したことが確認された場合、ステップ662に進み、そこで他のスレッドが関連付けられた動的リスト422にアクセスすることができるよう、関連付けられた動的リスト422をロック解除する、すなわちロック・アウト・ステップ658を逆転させる。ステップ662を実行した後、概念ロックCL_32はスレッド202によって解放されたと考えられ、他のスレッドが獲得することができる。

【0068】一方、図11のステップ660において、関連付けられた物理ロック420の内容がまだ概念ロックCL_32の識別表示でないこと、すなわち検査およびスワップ操作が再び失敗したことが確認された場合、ステップ664に進み、そこで概念ロックCL_32の識別表示を関連付けられた動的リスト422から除去する。概念ロックCL_32の識別表示は関連付けられた動的リスト422内になく、また関連付けられた物理ロック420の内容内になかった（とステップ660において判定された）ので、動的ロック構造要素M2は、ステップ664を実施した後、概念ロックCL_32の識別表示を追跡しない。したがって、後続のスレッドが現れ、図11のステップが終了した後、概念ロックCL_32を獲得することができる。

【0069】ステップ666、668、670、672および674において、可能なら、動的ロック構造要素M2がただ1つの概念ロック識別表示を記憶している場

合、概念ロック識別表示が好ましくは関連付けられた物理ロック420の内容内に移動されるよう、動的ロック構造要素M2を最適化する。したがって、残りの1つの概念ロックを保持するスレッドは、その概念ロックを解放したい場合、ステップ652、654および656を含むバスを介して迅速にそれを解放することができる。

【0070】ステップ666において、関連付けられた動的リスト422が、概念ロックCL_32の識別表示が（ステップ664において）関連付けられた動的リスト422から削除された後に残っているただ1つの概念ロック識別表示を有しているかどうかを確認する。関連付けられた動的ロック構造要素M2により追跡されるべき複数の概念ロックIDが残っている場合、ステップ674に進み、そこでスレッド202が概念ロックCL_32を解放するステップを終了する。

【0071】一方、動的ロック構造要素M2がその関連付けられた動的リスト422内にただ1つの概念ロック識別表示を追跡する場合、ステップ666からステップ668へ進み、そこで関連付けられた物理ロック420の内容を残りの1つの概念ロックの識別表示にセットする。ステップ670において、関連付けられた動的ロック構造要素M2がその物理ロック420内ならびにその動的リスト422内でこの残りの1つの概念ロックを追跡しないように、この残りの1つの概念ロックの識別表示を関連付けられた動的リストから削除する。

【0072】ステップ672において、後続のスレッドが関連付けられた動的リスト422にアクセスすることができるよう、関連付けられた動的リスト422をロック解除する、すなわちロックリスト・フラグ424をフラグ状態UNLOCK_LISTにセットすることによってロッキング・ステップ668を逆転させる。

【0073】本発明では、複数の動的リストを使用して、概念ロック識別表示を追跡することに留意された。さらに、その動的リストに関連付けられた動的ロック構造要素によって保持されている概念ロックが複数存在する場合、動的リストを使用する。代わりに单一の動的リストを使用して、保持されているすべての概念ロックを追跡した場合、スレッドは、所与のロックが現在保持されているかどうかを判定するために、またはそのリストから現在保持されているロックを削除するために、リスト全体を探査しなければならない。多数の概念ロックが保持されている場合、この探査には不适当に時間がかかり、またその実施が実行不可能になる。

【0074】本発明では、比較的少数の概念ロック（例えば、一実施形態では3つまたは4つ）を所与の動的ロック構造要素にマップする。したがって、動的リストによって保持される概念ロックの最大数は比較的小さくなり、したがって概念ロックの獲得または解放の速度が高くなる。さらに動的リストは、その動的リストに関連する動的ロック構造要素によって保持されている概念ロック

クが複数存在する場合にのみ使用される。動的ロック構造要素がただ1つの概念ロック識別表示を追跡する場合、物理ロック部分を使用して追跡を行うことが好ましく、それにより、本発明は、すべての記憶データ・オブジェクトについてロックを物理的に必要とする従来技術とほとんど同じ速さで概念ロックの獲得または解放を行うことができるようになる。

【0075】さらに、各動的ロック構造要素が平均してわずか数個または1つの概念ロックを保持するように動的ロック構造要素の最適数を選択することによって、コンピュータ・システムの性能を調整することができる。各動的ロック構造要素が平均してわずか1つの概念ロック識別表示を保持できるようにする（したがって物理ロックを最大限に使用する）ことによって性能を最適化する実施形態では、コンピュータ設計者は、所与の時点においてアクセスすべき記憶データ・オブジェクトの平均数にはば比例するように、動的ロック構造要素の数を選択することができる。この所与の時点においてアクセスすべき記憶データ・オブジェクトの平均数は、例えば、統計的方法または他の従来の方法によって試行錯誤して決定することができる。例えば、ある所与の時点において使用できる概念ロックの10分の1しか保持されていないことが確認された場合、動的構造は、(N/10)個の動的ロック構造要素またはその整数倍を有しており（ただしNは、概念ロックの総数を表す）、したがって動的ロック構造要素内、すなわちその物理ロック部分内に平均してただ1つの概念ロック識別表示が記憶されることになる。したがって、アクセス速度が有利に最大になる。一実施形態では、設計者は、動的ロック構造要素の数を減らすことによって、速度とメモリ・オーバヘッドとの兼ね合いをはかることができる。動的ロック構造要素が平均して複数の概念ロック識別表示を保持している場合でもなお、従来技術のメモリ・オーバヘッドを減らす目的は有利に達成される。

【0076】さらに、本発明は、本願に記載のロック以外のタイプのロックを使用するシステムにも十分同様に適用できる。あるシステムでは、例えば、スレッドが読み取りまたは書き込みのために記憶データ・オブジェクトにアクセスする場合、その記憶データ・オブジェクト上のロックが獲得される。他のシステムでは、複数のスレッドが記憶データ・オブジェクトを同時に読み取ることはできるが、その記憶データ・オブジェクトに書き込みを行

いたいスレッドがロックを獲得することはできないロックが存在する。本願に記載の発明を異なるタイプのロックに適合させることは、この開示を得た当業者の力量でできることである。

【0077】以上、本発明についていくつかの好ましい実施形態に関して説明したが、本発明の範囲内に入る改変例、置換例および同等物もある。また、本発明の方法および装置を実施する他の方法も多数ある。したがって、首記の請求の範囲は、本発明の真の精神および範囲内に入るすべての改変例、置換例および同等物を含むものと解釈すべきものである。

【図面の簡単な説明】

【図1】複数のスレッドがプロセッサを介して多重方式で実行される多重プログラミング状況を示す図である。

【図2】複数のスレッドが複数のプロセッサ上で実行される2つの異なる多重処理状況を示す図である。

【図3】複数のスレッドが複数のコンピュータ上で実行される2つの異なる多重処理状況を示す図である。

【図4】複数のスレッドが単一の記憶データ・オブジェクトへのアクセスを許される場合に生じることあるアクセス競合を回避する従来の技術を示す図である。

【図5】本発明の一態様に従ってアクセス競合を回避するために使用される構造を示す図である。

【図6】本発明の一態様に従って図5の動的ロック構造を詳細に示す図である。

【図7】一例において図5の動的ロック構造の各要素中に存在する各種パラメータを示す図である。

【図8】一実施形態において記憶データ・オブジェクトにアクセスするために概念ロックを獲得したい場合に行うステップを示す流れ図である。

【図9】一実施形態において記憶データ・オブジェクトにアクセスするために概念ロックを獲得したい場合に行うステップを示す流れ図である。

【図10】一実施形態において概念ロックを解放したい場合に行うステップを示す流れ図である。

【図11】一実施形態において概念ロックを解放したい場合に行うステップを示す流れ図である。

【符号の説明】

202、204、206 スレッド

40 310、312、314、320、322、324 記憶データ・オブジェクト

350 動的ロック構造

【図1】

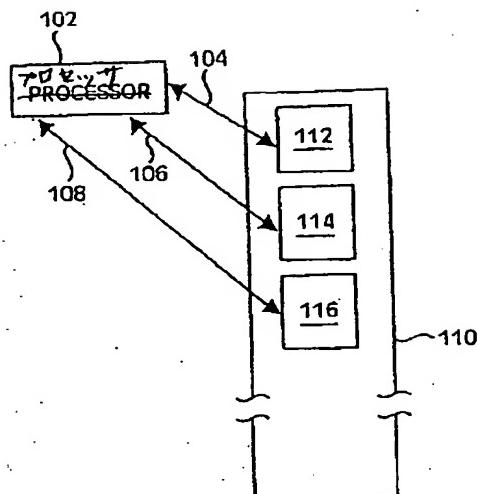


FIG. 1A

【図2】

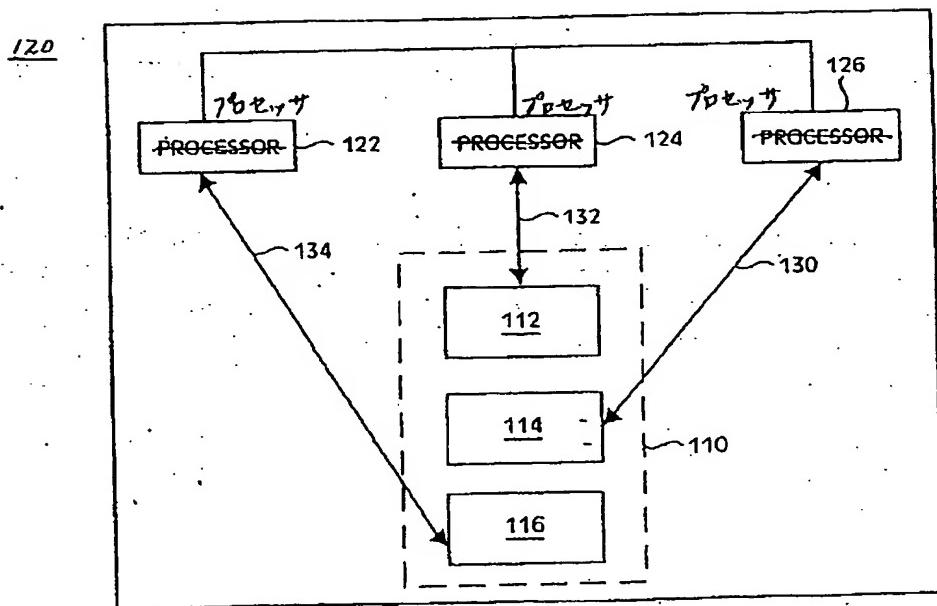


FIG. 1B

【図3】

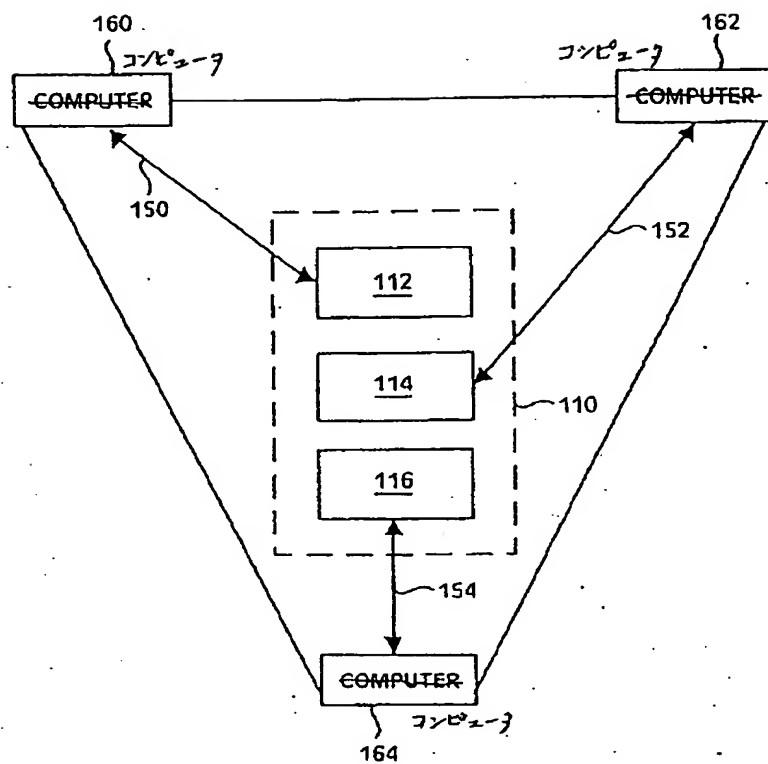
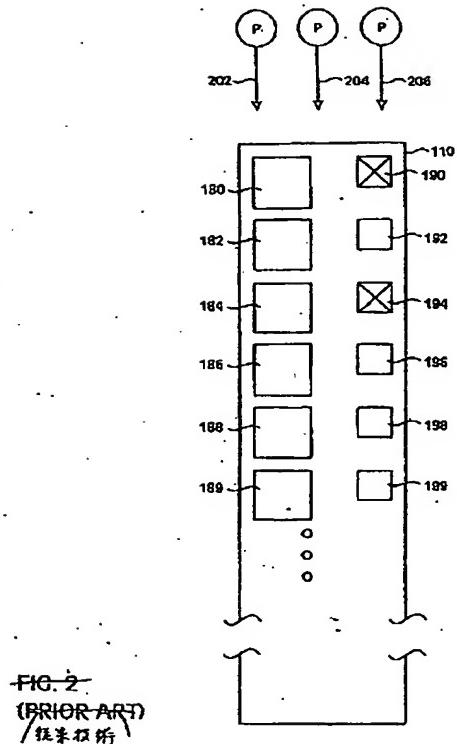
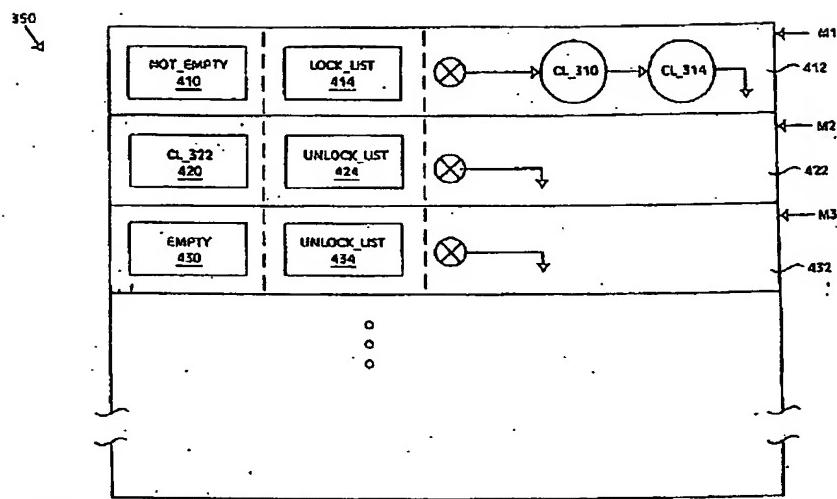


FIG. 1C

【図4】



【図7】



【図5】

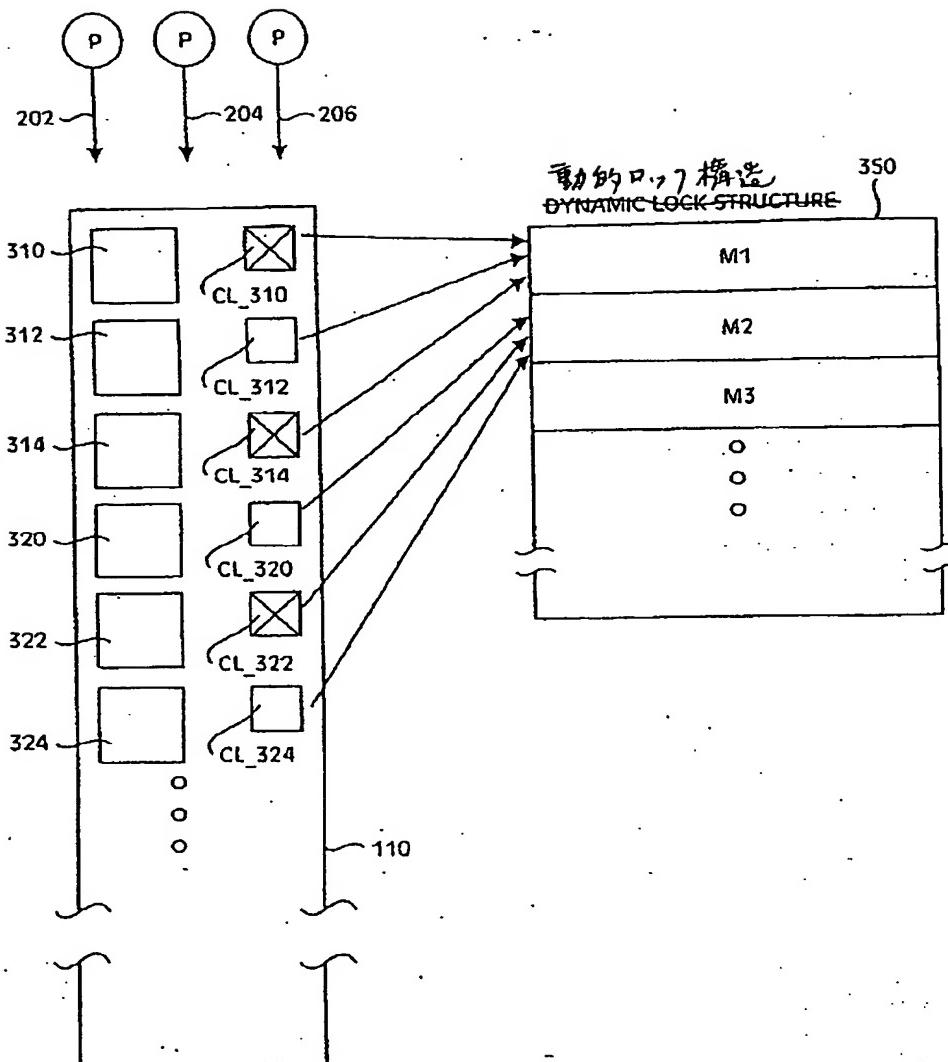


FIG. 3

【図6】

物理ロック (PHYSICAL LOCK) <u>370</u>	リストロックフラグ (LIST-LOCK FLAG) <u>372</u>	動的リスト (DYNAMIC LIST) <u>374</u>	M1
物理ロック (PHYSICAL LOCK) <u>380</u>	リストロックフラグ (LIST-LOCK FLAG) <u>382</u>	動的リスト (DYNAMIC LIST) <u>384</u>	M2
物理ロック (PHYSICAL LOCK) <u>390</u>	リストロックフラグ (LIST-LOCK FLAG) <u>392</u>	動的リスト (DYNAMIC LIST) <u>394</u>	M3
○	○	○	

350
↑

FIG. 4A

[図8]

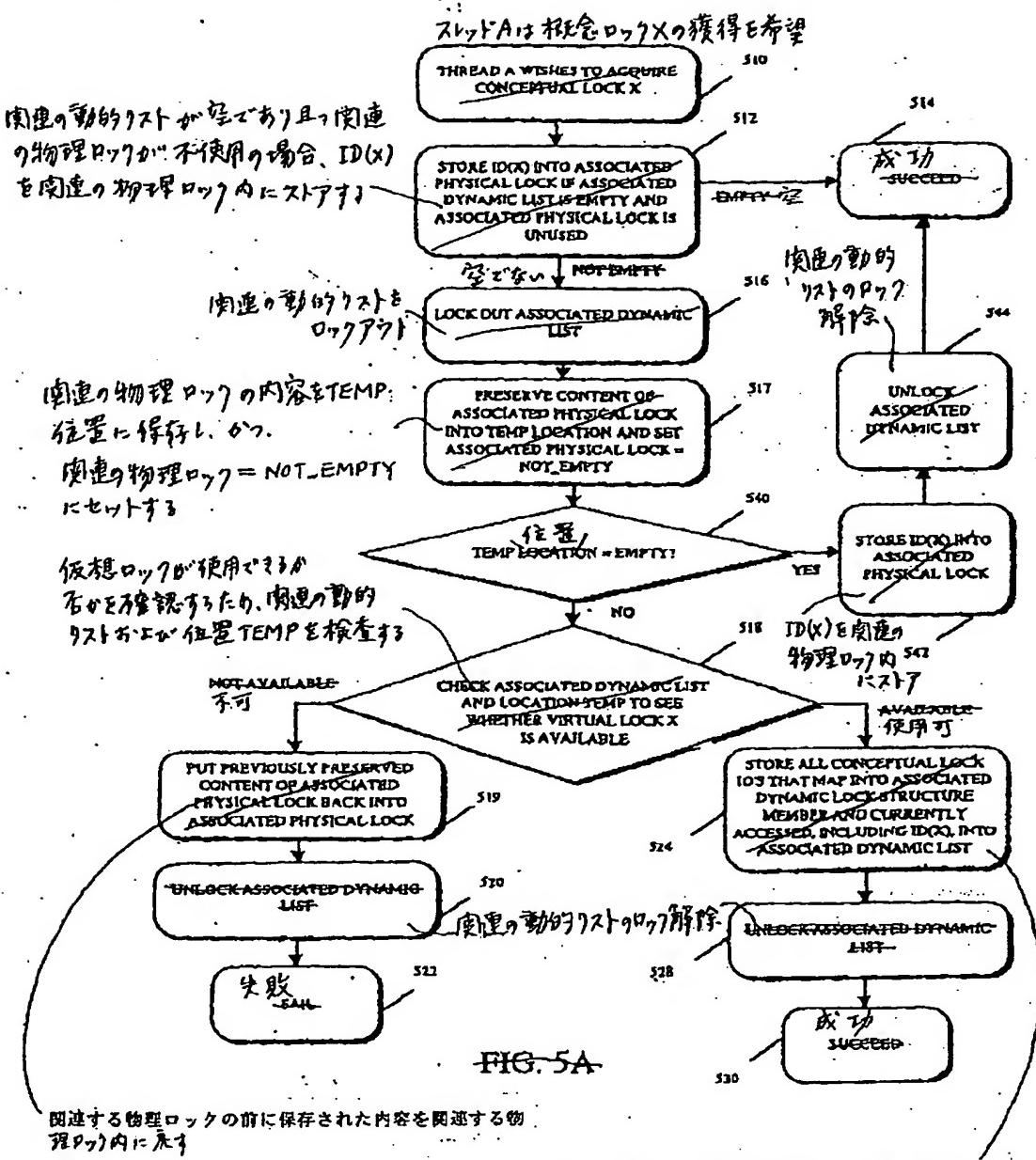
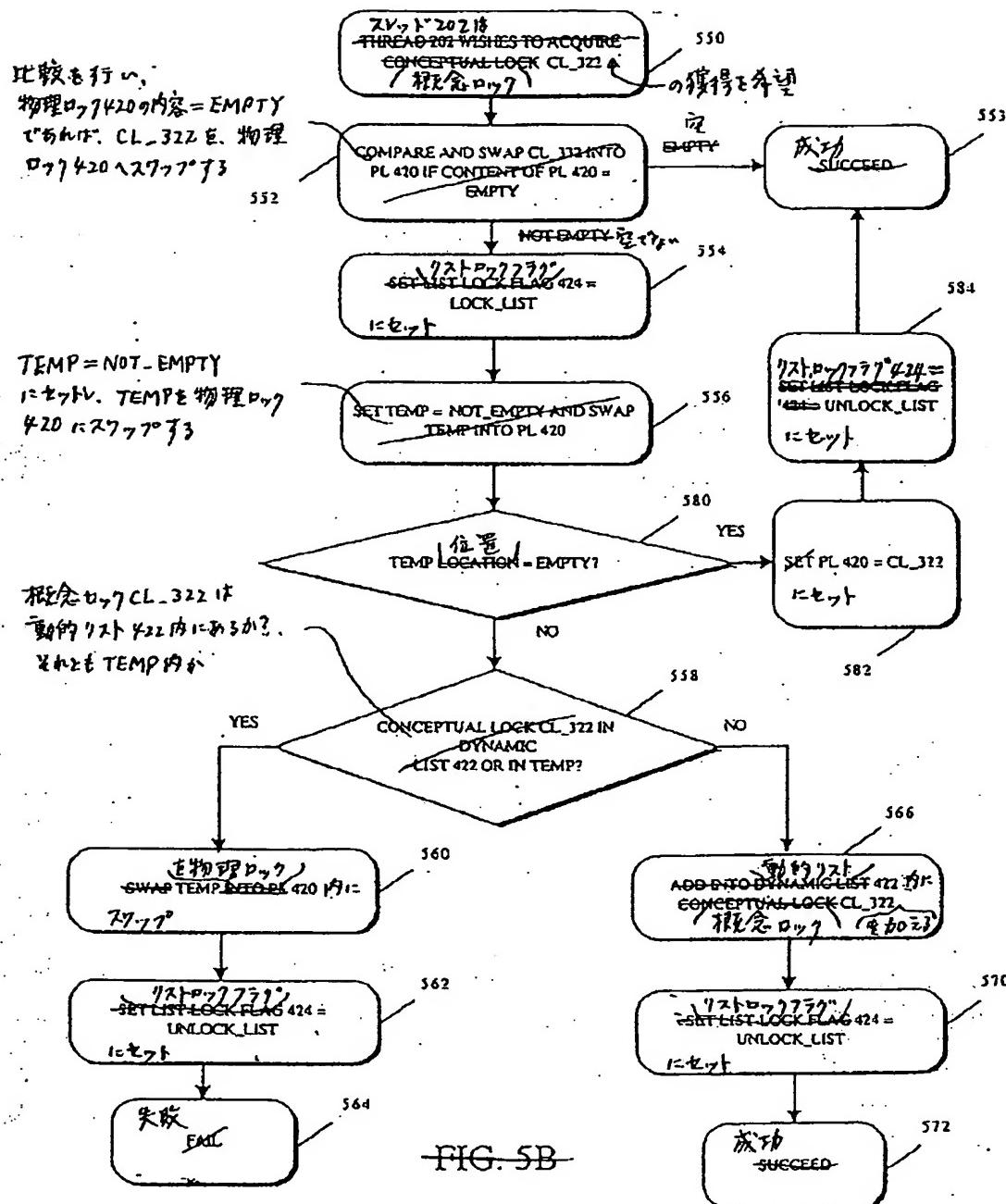
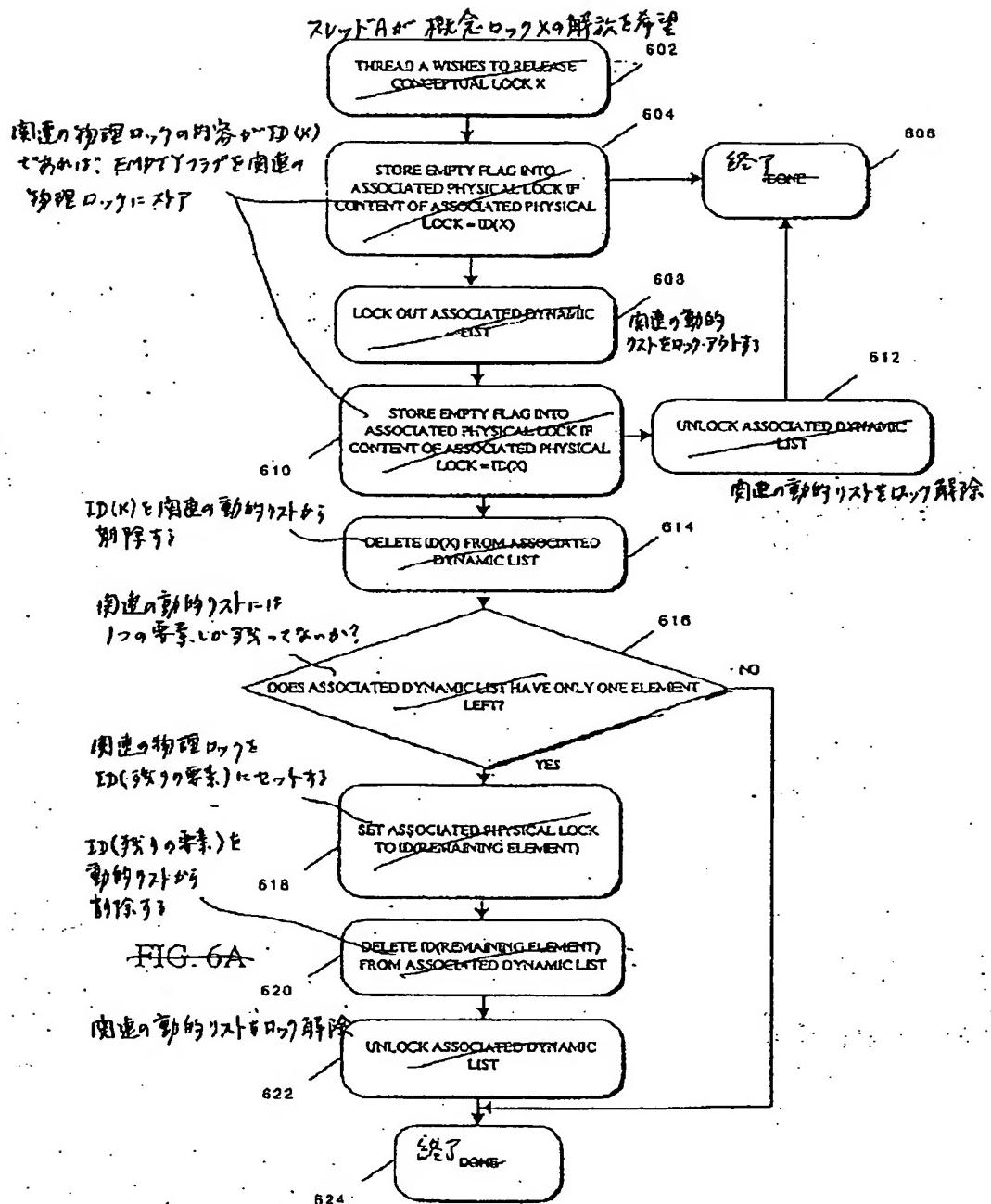


FIG. 5A

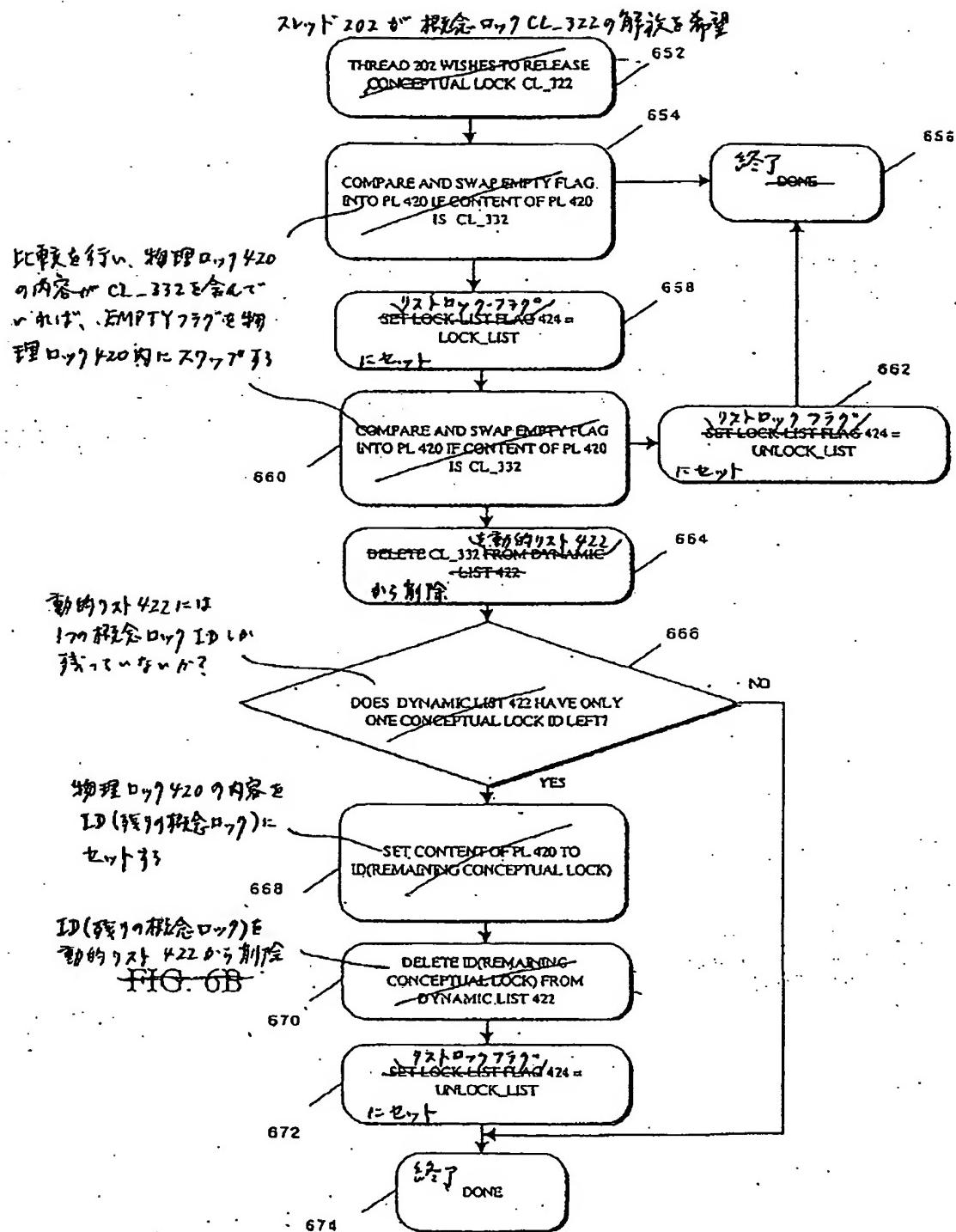
【図9】



【図10】



【図11】



(22)

特開平10-187527

フロントページの続き

(71)出願人 591064003
901 SAN ANTONIO ROAD
PALO ALTO, CA 94303, U.
S. A.

(72)発明者 マーク・ドナルド・ヒル
アメリカ合衆国・53705・ウィスコンシン
州・マディソン・チャンバーレイン アヴ
エニュ・2124

【公報種別】特許法第17条の2の規定による補正の掲載

【部門区分】第6部門第3区分

【発行日】平成11年(1999)7月2日

【公開番号】特開平10-187527

【公開日】平成10年(1998)7月21日

【年通号数】公開特許公報10-1876

【出願番号】特願平9-187269

【国際特許分類第6版】

G06F 12/00 572

9/46 340

【F I】

G06F 12/00 572 A

9/46 340 F

【手続補正書】

【提出日】平成10年3月23日

* 【補正対象項目名】全図

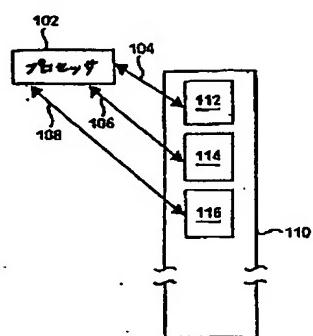
【手続補正1】

【補正方法】変更

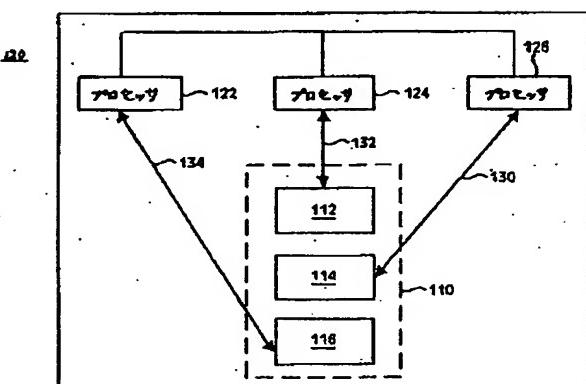
【補正対象書類名】図面

* 【補正内容】

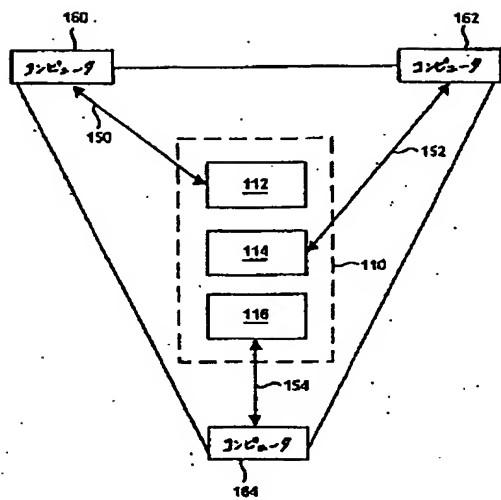
【図1】



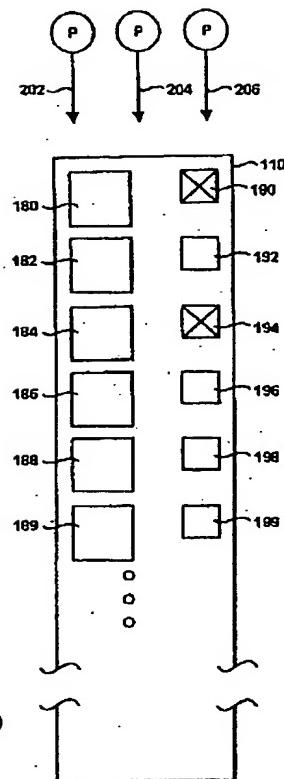
【図2】



【図3】



【図4】

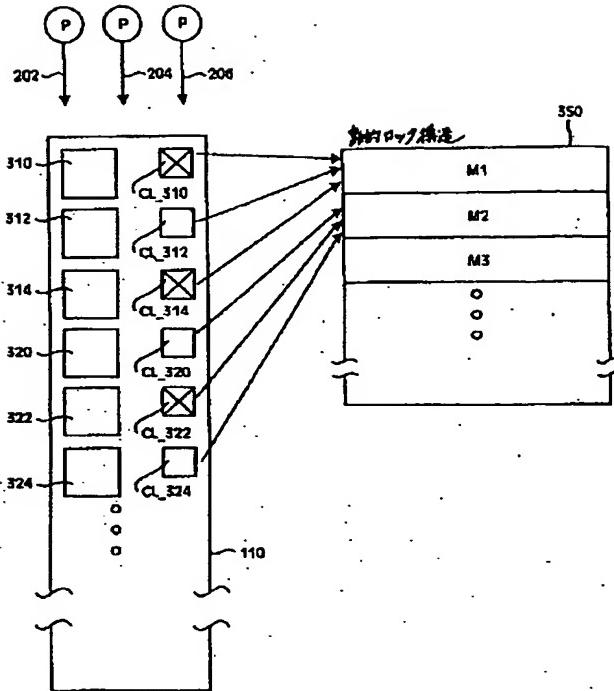


【図6】

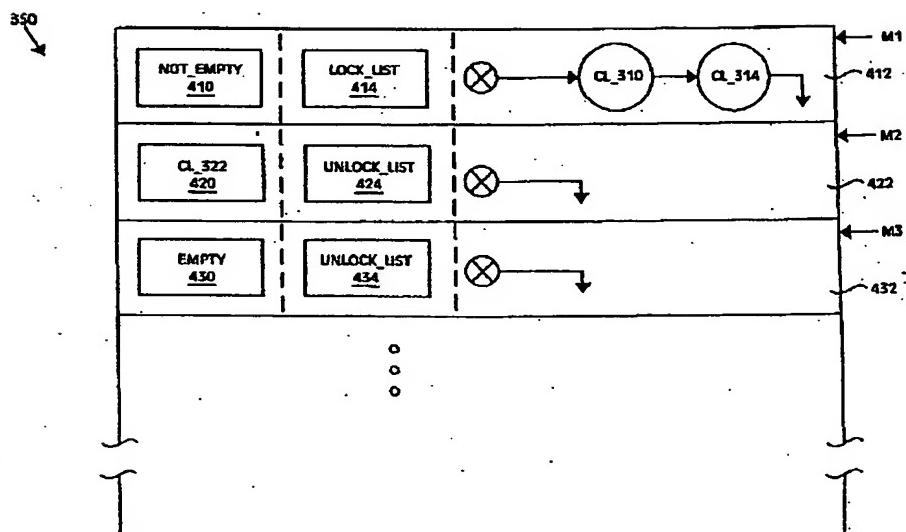
物理ロック (PHYSICAL LOCK) 370	リストロックフラグ (LIST-LOCK FLAG) 372	動的リスト (DYNAMIC LIST) 374
物理ロック (PHYSICAL LOCK) 380	リストロックフラグ (LIST-LOCK FLAG) 382	動的リスト (DYNAMIC LIST) 384
物理ロック (PHYSICAL LOCK) 390	リストロックフラグ (LIST-LOCK FLAG) 392	動的リスト (DYNAMIC LIST) 394
○ ○ ○		

350

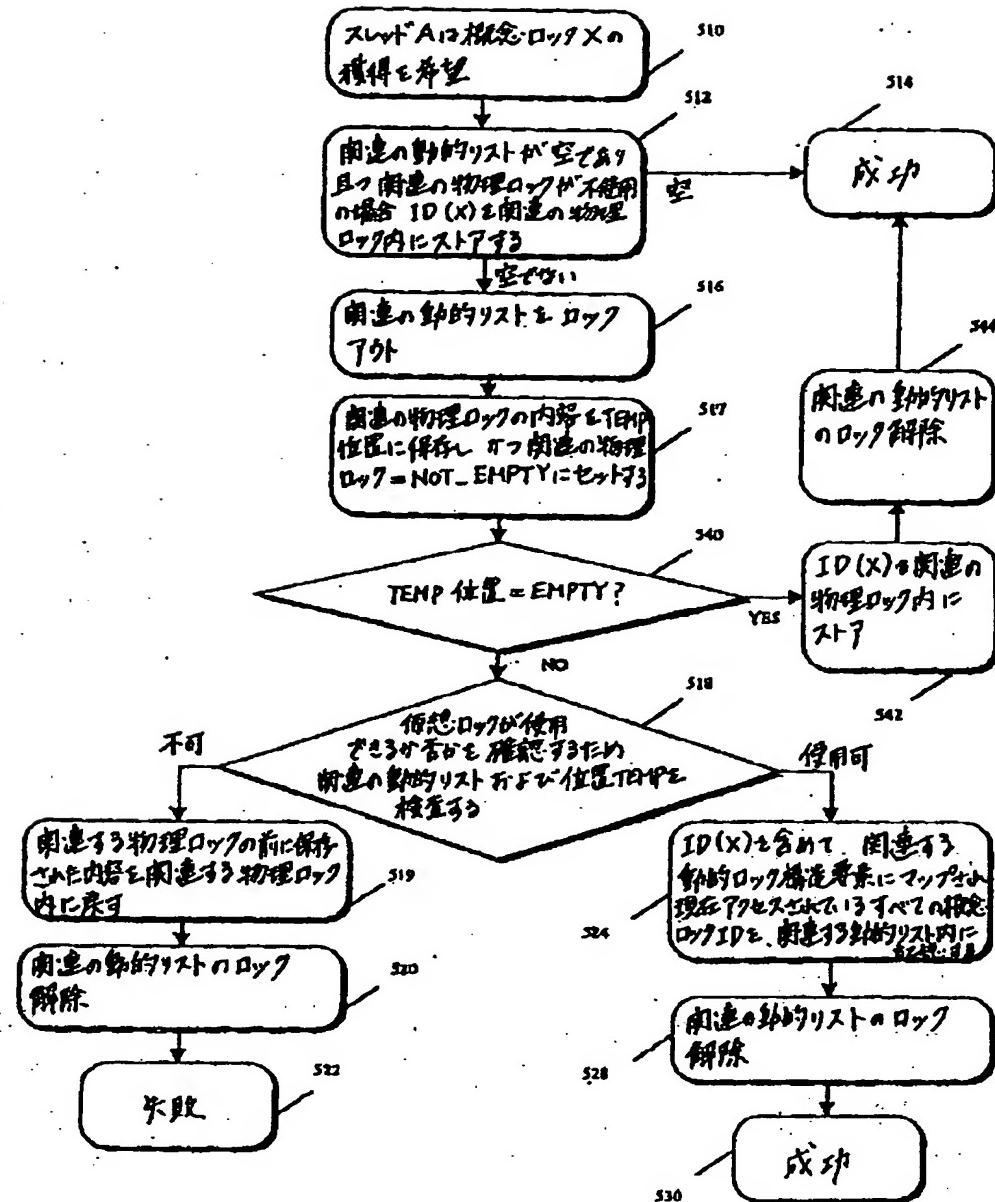
【図5】



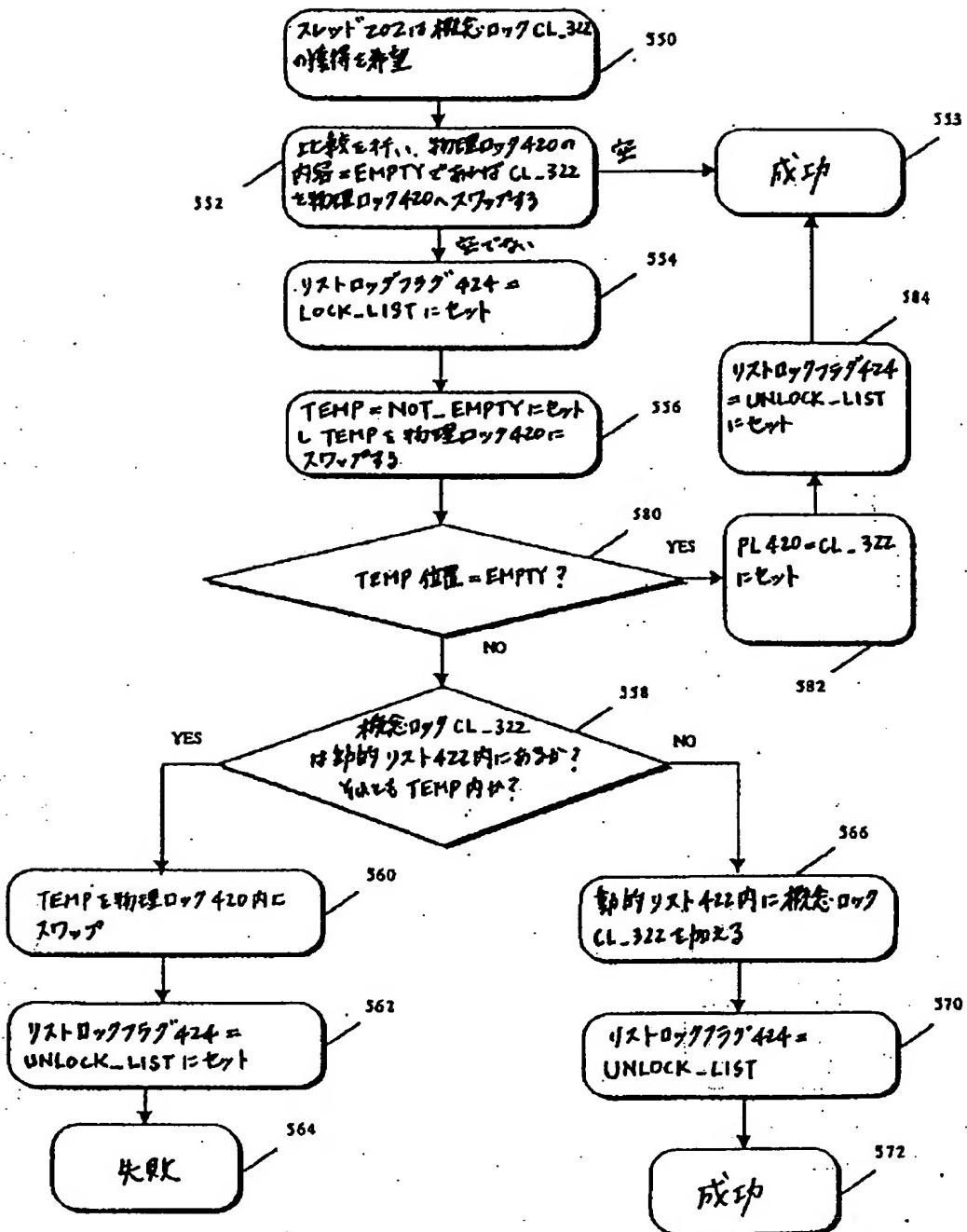
【図7】



【図8】



【図9】



【図10】

